TYLER: Testing Data Set and Results for ML models

### PREPARED FOR

TechE, TechnoUtsav 3.0 2020

Deloitte

### PREPARED BY

SPIT Mumbai\_Deloitteful Devs

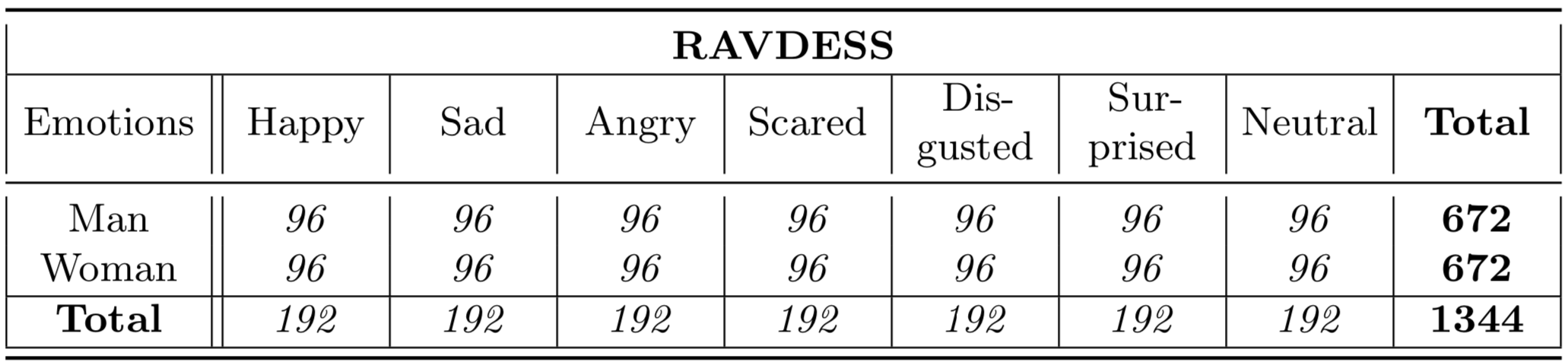
1. Jimil Shah
2. Rahul Soni
3. Dipam Shah

### Time-Distributed CNN-LSTM Model for MultiModal Emotion Recognition

**Dataset**

The data set used for training is the Ryerson Audio-Visual Database of Emotional Speech and Song: <https://zenodo.org/record/1188976/files/Audio_Speech_Actors_01-24.zip?download=1>

RAVDESS contains 24 professional actors, vocalizing two lexically matched statements in a neutral North American accent. Speech includes calm, happy, sad, angry, fearful, surprise, and disgust expressions. Each expression is produced at two levels of emotional intensity (normal, strong), with an additional neutral expression. Save the audio files in the **DATAS/RAVDESS** directory in the root. Ensure that the Actor subfolders are removed, and data is placed directly under the **DATAS/RAVDESS** folder.



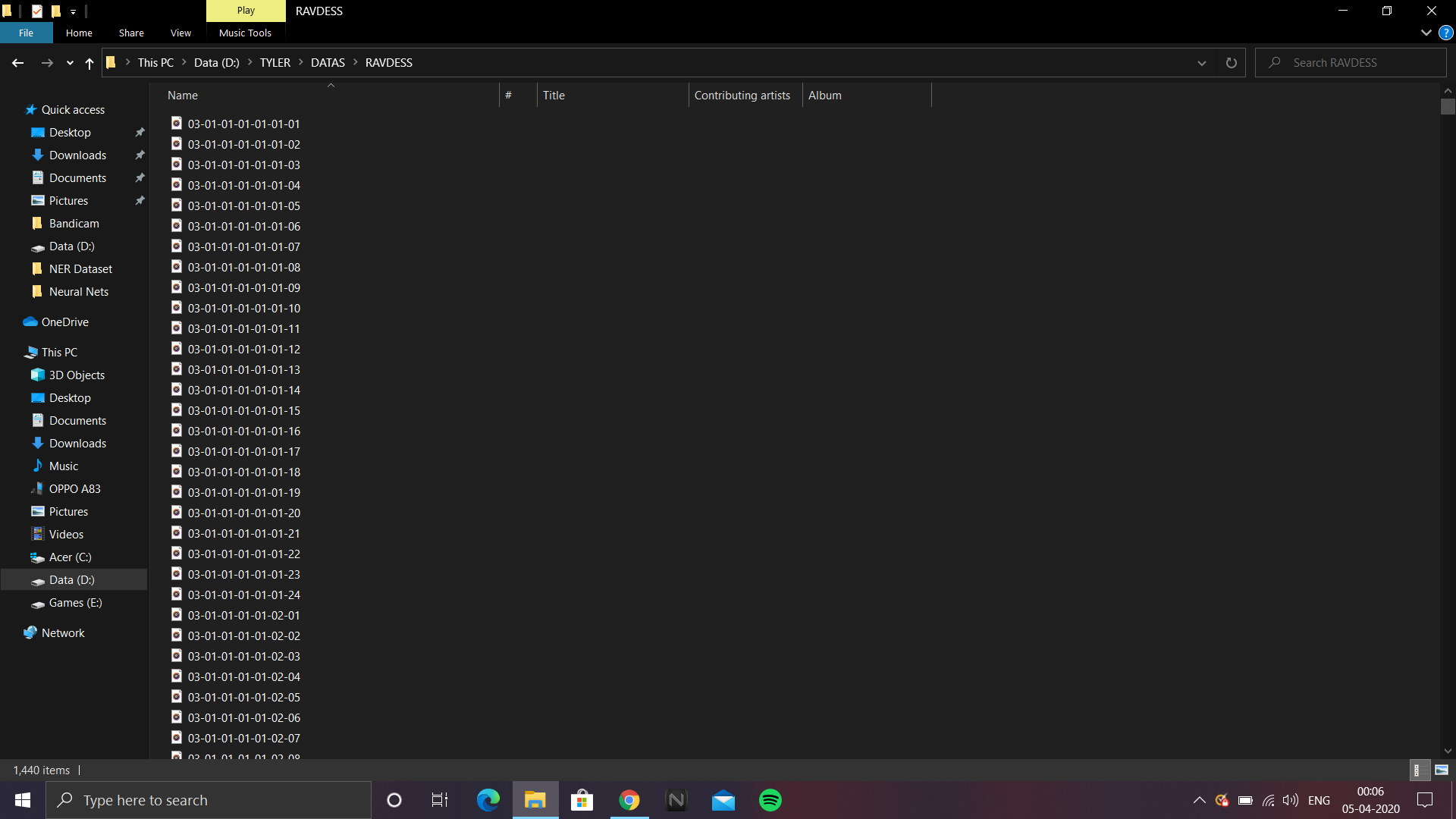
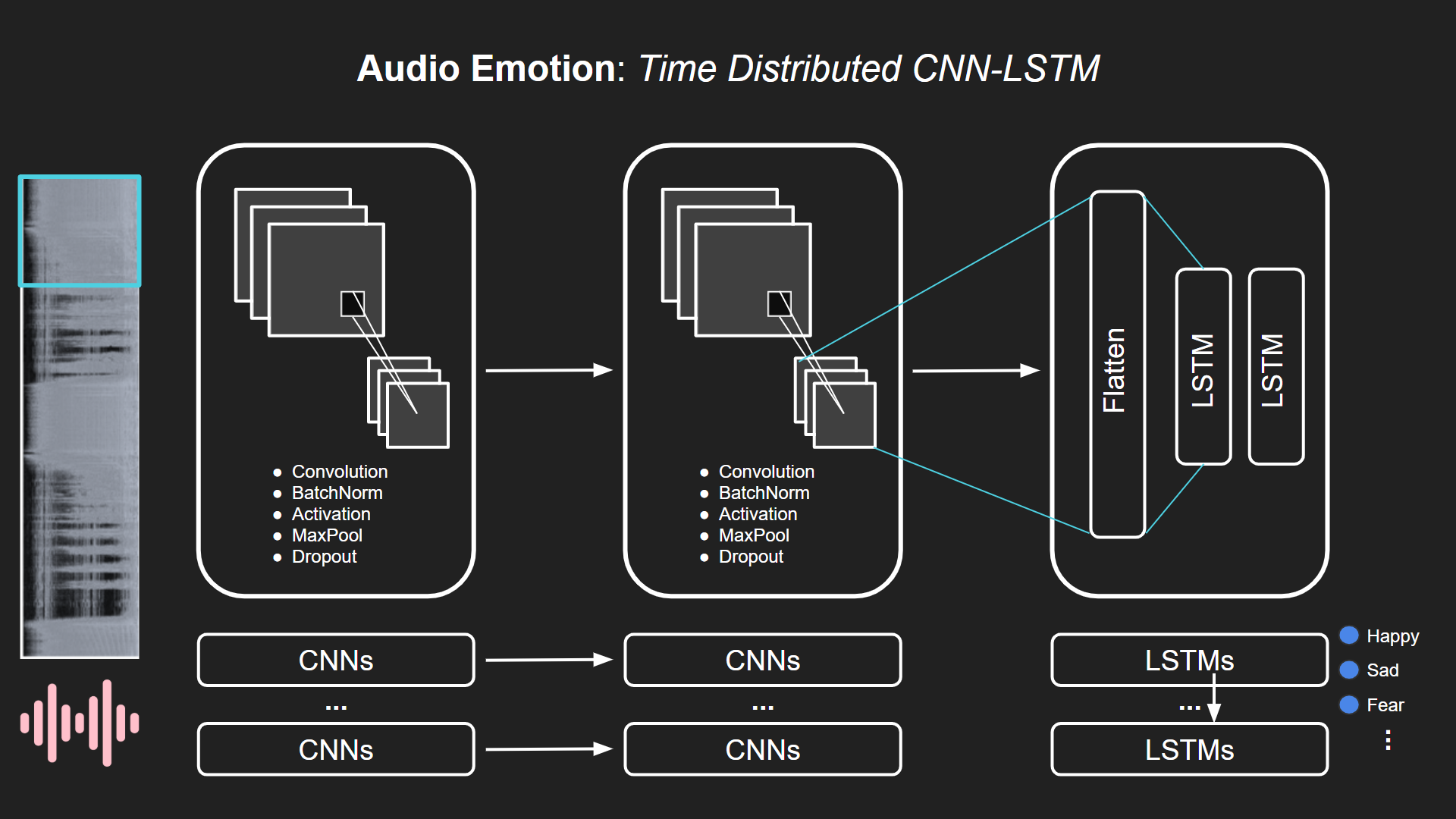


Fig. DATAS/RAVDESS Directory

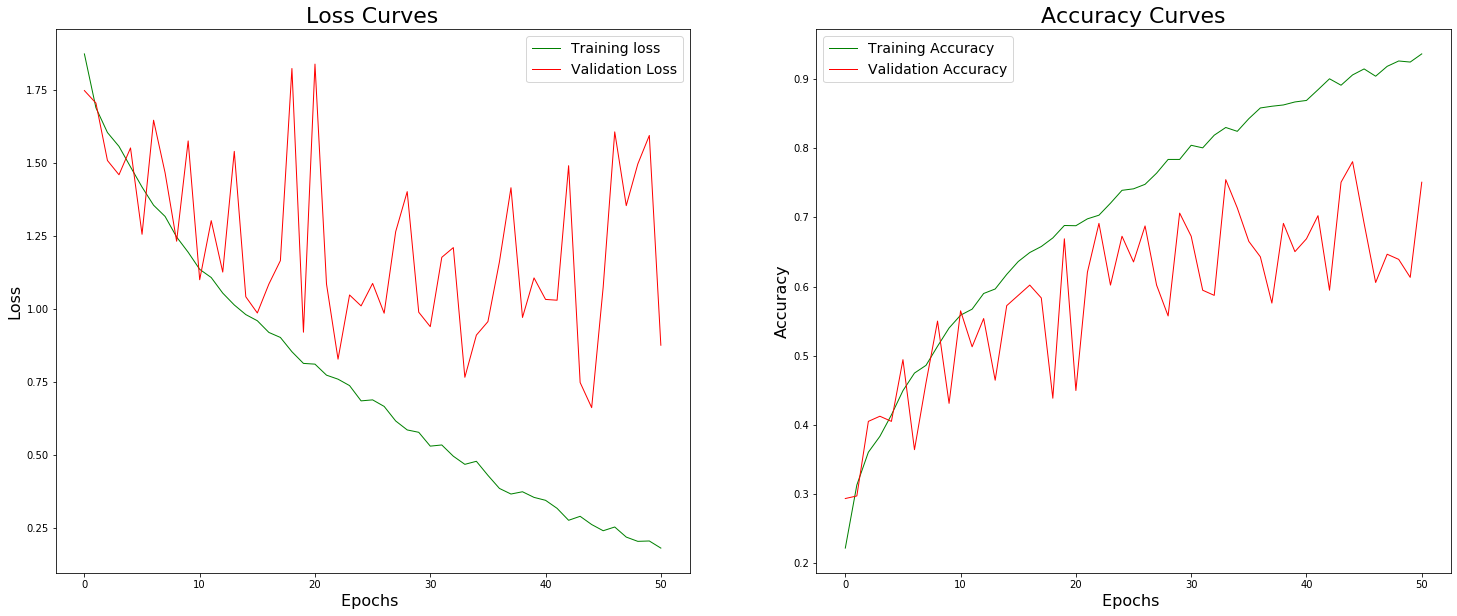
**Model Architecture**

The main idea of a Time Distributed Convolutional Neural Network is to apply a rolling window (fixed size and time-step) all along the log-mel-spectrogram. Each of these windows will be the entry of a convolutional neural network, composed by four Local Feature Learning Blocks (LFLBs) and the output of each of these convolutional networks will be fed into a recurrent neural network composed by 2 cells LSTM (Long Short Term Memory) to learn the long-term contextual dependencies. Finally, a fully connected layer with softmax activation is used to predict the emotion detected in the voice.



**Training the Model**

View the Jupyter notebook [MultiModal Emotion Recognition using CNN-LSTM.ipynb] in the root directory of the executable code to see the model training and testing results. After 50 epochs, the model stopped training with a validation accuracy of 75%. This is visualized in the below plots:



**Testing the Model**

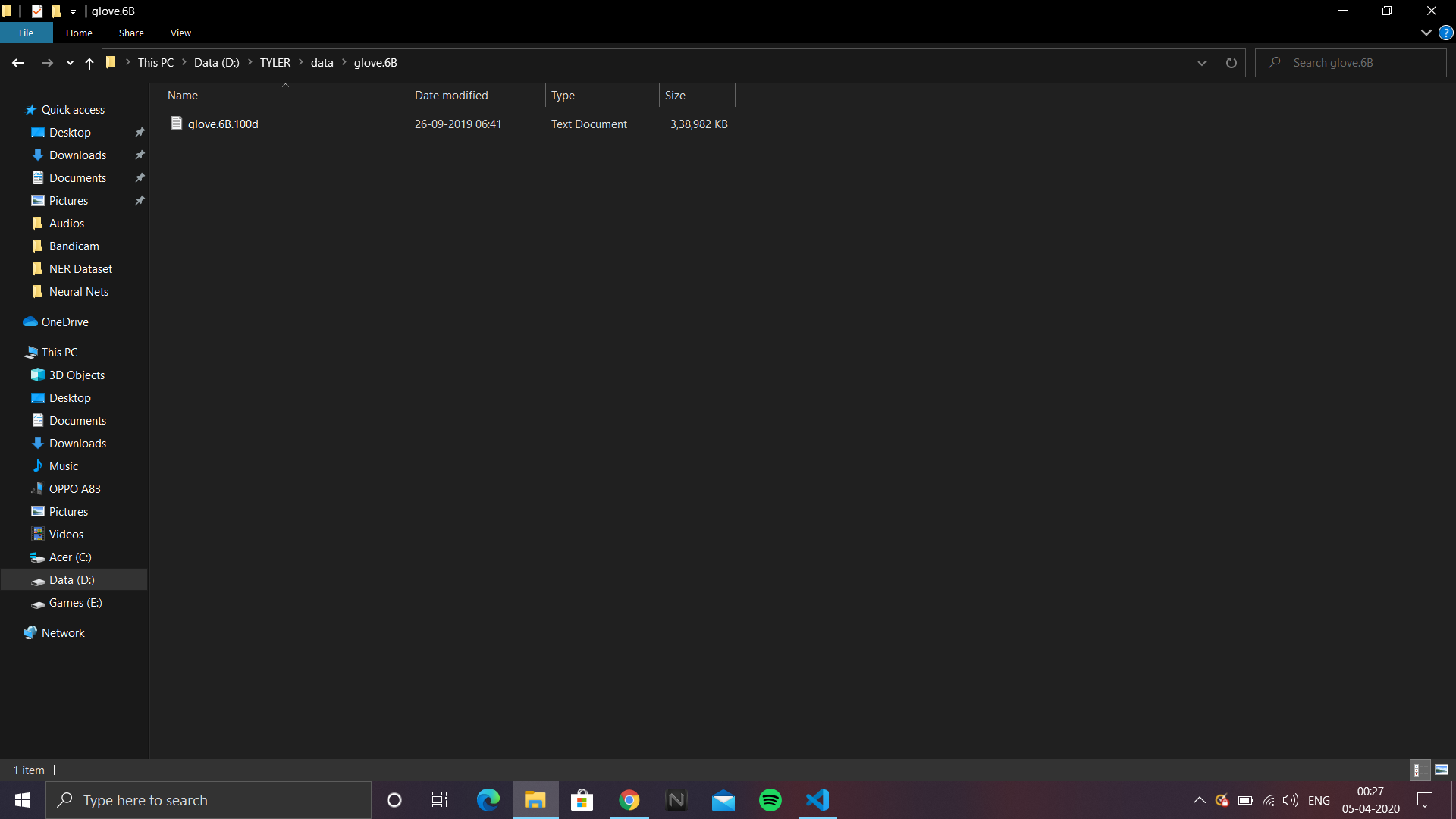
Once the model training is complete, the model and it’s weights are saved under the Models/ subdirectory. The trained model can also be cloned or downloaded from [this GitHub repository](https://github.com/JimilProgGrammer/tyler-models). The models to be used are [CNN-LSTM]Model.h5 and [CNN-LSTM]Weights.h5. You can then run the below command in an Anaconda Prompt:

|  |
| --- |
| D:\TYLER> python test\_audio\_emotion.py |

### Bi-LSTM+CRF Model for Clinical Named Entity Recognition (NER)

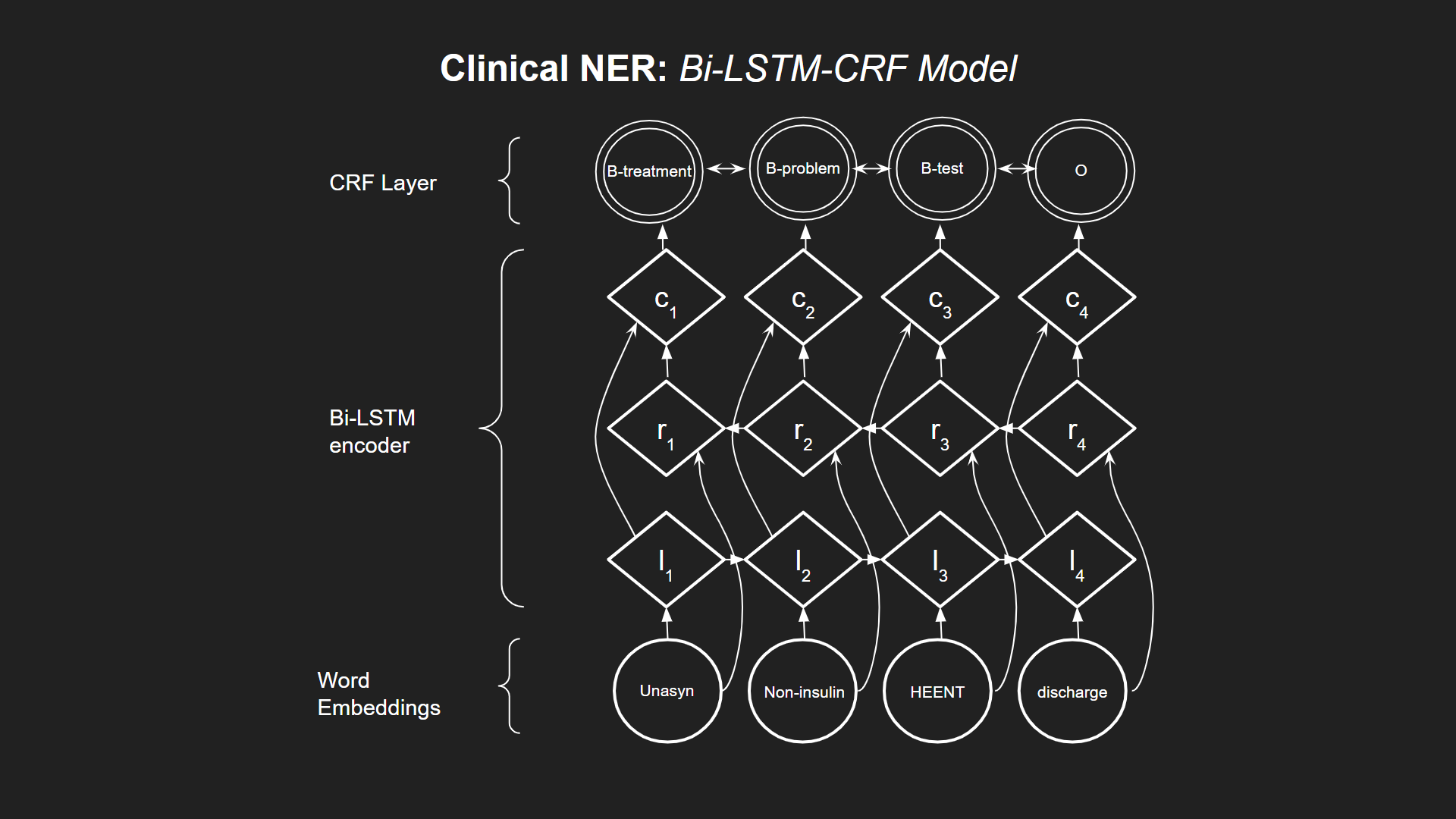
**Dataset**

The model is trained on the extensive i2b2 dataset. The i2b2 (Informatics for Integrating Biology & the bedside) dataset has provided fully de-identified notes from the Research Patient Data Repository at Partners HealthCare for a series of NLP Challenges organized by Dr. Ozlem Uzuner. The model also requires downloading the pre-trained Stanford GloVE embeddings. GloVe is an unsupervised learning algorithm for obtaining vector representations for words. i2b2 train, dev, and test data has been provided along with the uploaded code. The GloVE embeddings can be downloaded from: <https://www.kaggle.com/terenceliu4444/glove6b100dtxt/downloadand> must be placed under **data/glove.6B** directory under the root folder.



**Model Architecture**

We train our networks using the back-propagation algorithm updating our parameters on every training example, one at a time, using stochastic gradient descent (SGD) a learning rate of 0.01 and a gradient clipping of 5.0. Our LSTM-CRF model uses a single layer for the forward and backward LSTMs whose dimensions are set to 100. We set the dropout rate to 0.5. Using higher rates negatively impacted our results, while smaller rates led to longer training time. [[Reference](https://arxiv.org/pdf/1603.01360.pdf)]



**Training the Model**

To train the model, run the training.py file under the root directory. Once the training is complete, the file automatically saves the best model weights and params under the **Models** subfolder in the root directory. This model can further be used for tagging purposes.

**Testing the Model**

Once the training is completed, the model weights along with parameters are saved under the Models/ subdirectory. You can also clone or download the trained model weights from [this Github repository](https://github.com/JimilProgGrammer/tyler-models), Download the model\_weights.h5, params.json and preprocessor.json files and place them under the Models/ subdirectory. You can then test the model by running the below command in an Anaconda Prompt:

|  |
| --- |
| D:\TYLER> python test\_clinical\_ner.py |

### LSTM+Word Embeddings Model for Cognitive Behavioural Therapy Chatbot

**Dataset**

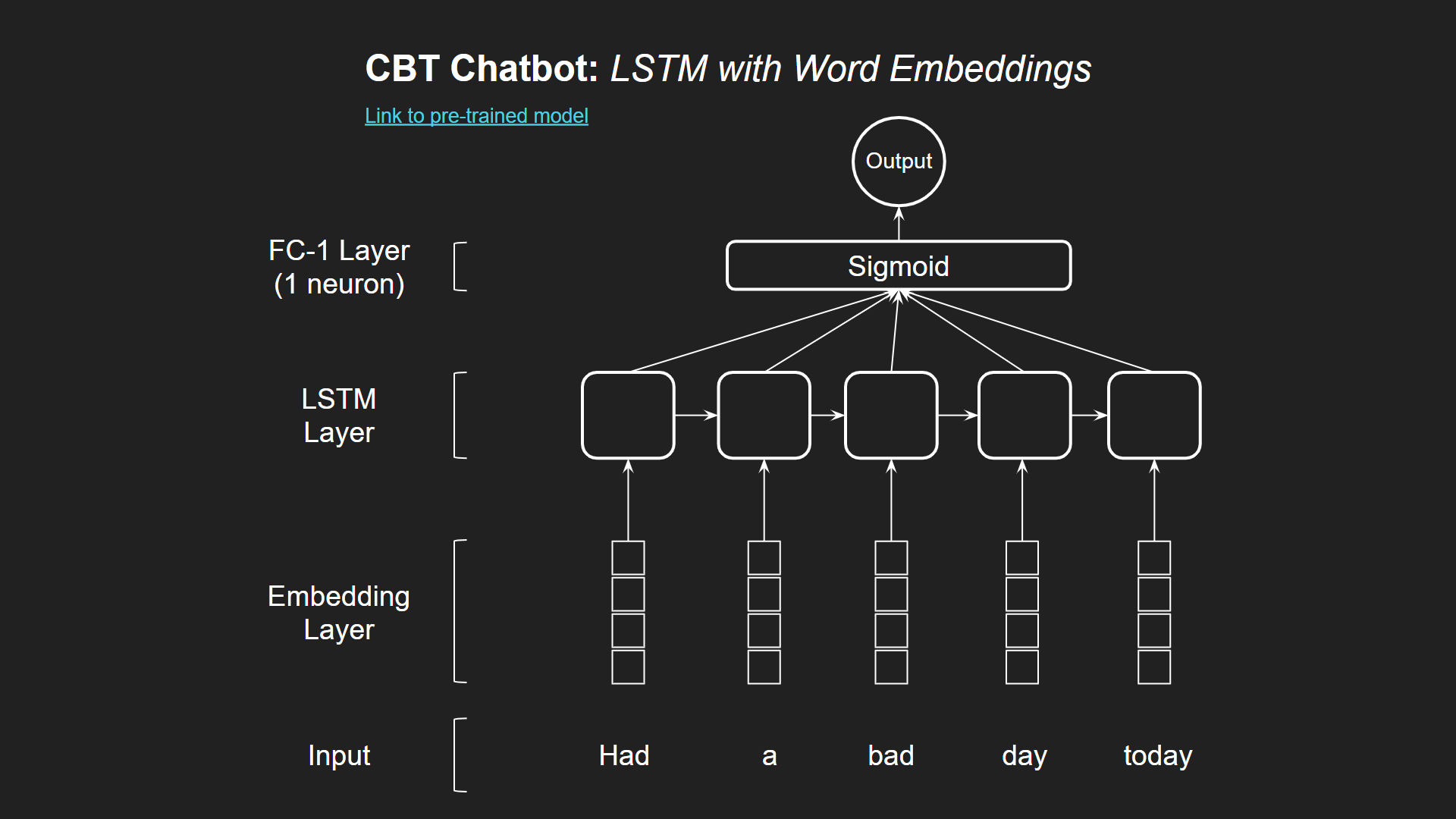
1.6 million tweets from twitter, classified into positive (4) and negative (0), as the training data, of which 20% is used as cross validation data. 494 tweets from twitter, classified into positive (4), neutral (2) and negative (0), as the test data. Labels adjusted to be in the range 0 to 1.

Hashtags, website links and user references removed, then input tweets preprocessed by Gensim, with preprocessed tweets of length less than two removed. Vocabulary of words initialised with Gensim Dictionary and words replaced with respective position in the vocabulary plus one.

Preprocessed tweets of length less than 20 were zero-padded to length 20. Those of length greater than 20 were split into tweets of length 20 and the last split part zero-padded, if necessary. Zero Padding done for supplying variable length sequences to the LSTM layer.

**Model Architecture**

An Embedding layer with zero-masking to output word vectors of 32 dimensions for each word in the vocabulary, and a zero vector for zero-padded words. This is followed by an LSTM layer with 128 dimensional output. Output is a fully connected output layer with 1 neuron and sigmoid activation function. The neural network uses binary cross-entropy loss function and the Adam optimizer with default parameters, but with Nesterov momentum.



**Training the Model**

A pre-trained model from [this repo](https://github.com/rharish101/PClub-Project) was used to develop the chatbot. Dataset for learning was obtained from <http://help.sentiment140.com/for-students/.> Current results showed accuracy of 84.14% on the training data, accuracy of 83.32% on the cross validation data and accuracy of 60.17% on the test data.

**Testing the Model**

You can have a chat with the chatbot by running the bot.py file under the root directory using the following command in an Anaconda Prompt:

|  |
| --- |
| D:\TYLER> python bot.py |

It uses the LSTM + Word Embeddings Model to assign a numerical score for each text message that you send to the bot. This numerical value is then used to direct conversation and understand how you are feeling.

### GitHub Links for the resulting trained models and weights

All of the above 3 models and their weights are also uploaded to GitHub for easier access. They can be found on the below link:

<https://github.com/JimilProgGrammer/tyler-models>

The contents of the repository are as described in the below table. Clone or download this repository and place all the contents in the Models subdirectory under the project’s root, if you want to directly test the application without training the models.

|  |  |
| --- | --- |
| Bot\_Model.h5  Bot\_Vocab\_Sentiment | Neural Network model for CBT chatbot. |
| [CNN-LSTM]Model.h5  [CNN-LSTM]Weights.h5 | Neural Network model to predict emotion from submitted audio recording using log-mel spectrogram. |
| model\_weights.h5  params.json  preprocessor.json | Neural network model for detecting symptoms from text records. |